AD-A103 096   KANSAS STATE UNIV MANHATTAN DEPT OF COMPUTER SCIENCE   F/6 9/2
RESEARCH IN FUNCTIONALLY DISTRIBUTED COMPUTER SYSTEMS DEVELOPME--ETC(U)
JAN 77 F J MARYANSKI, V E WALLENTINE   DAAG29-76-G-0108

UNCLASSIFIED CS-77-1   NL

1 OF 1
AD A
03096

END
DATE
FILMED
9-81
DTIC

15

AIRMICS

Army Institute for Research in
Management Information and
Computer Science

313 Calculator Bldg.
GA Institute of Technology
Atlanta, GA 30332

Technical Report

RESEARCH IN FUNCTIONALLY DISTRIBUTED COMPUTER SYSTEMS DEVELOPMENT

Kansas State University

Virgil Wallentine
Principal Investigator

Approved for public release; distribution unlimited

VOLUME XI

A SURVEY OF DEVELOPMENTS IN DISTRIBUTED DATA BASE MANAGEMENT SYSTEMS

U.S. ARMY COMPUTER SYSTEMS COMMAND FT BELVOIR, VA 22060

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)   (11) Interim rept.

REPORT DOCUMENTATION PAGE		READ INSTRUCTIONS BEFORE COMPLETING FORM
~~1. REPORT NUMBER~~ (6) Research Efforts / Distributed Computer System Developments, Volume II	2. GOVT ACCESSION NO. AD-A103096	3. RECIPIENT'S CATALOG NUMBER
4. TITLE (and Subtitle) A SURVEY OF DEVELOPMENTS IN DISTRIBUTED DATA BASE MANAGE[redacted] SYSTEMS. ment		5. TYPE OF REPORT & PERIOD COVERED Iterim
		6. PERFORMING ORG. REPORT NUMBER (14) CS-77-1
7. AUTHOR(s) Fred Maryanski		8. CONTRACT OR GRANT NUMBER(s) (15) ✓DAAG 29-76-G-0108
9. PERFORMING ORGANIZATION NAME AND ADDRESS Kansas State University Department of Computer Science Manhattan, KS 66506		10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS (11)
11. CONTROLLING OFFICE NAME AND ADDRESS US Army Research Office P O Box 12211 Research Triangle Park, NC 27700		12. REPORT DATE / Jan[redacted]1977 /
		13. NUMBER OF PAGES 36 pages
14. MONITORING AGENCY NAME & ADDRESS (if different from Controlling Office) US Army Computer Systems Command Attn: CSCS-AT Ft. Belvoir, VA 22060 (12) 40		15. SECURITY CLASS. (of this report) Unclassified
		15a. DECLASSIFICATION DOWNGRADING SCHEDULE

16. DISTRIBUTION STATEMENT (of this Report)

Approved for public release; distribution unlimited.

17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report)

18. SUPPLEMENTARY NOTES

The findings in this report are not to be construed as an official Department of the Army position, unless so designated by other authorized documents.

19. KEY WORDS (Continue on reverse side if necessary and identify by block number)

DBMS
DDBMS
Distributed Processing

20. ABSTRACT (Continue on reverse side if necessary and identify by block number)

-over-

39112

-ABSTRACT-

Recently we have witnessed the advent of general purpose data base management systems and important advances in computer networks. The combination of the two technologies to produce distributed data base management systems should be the next significant step in commercial systems development. A completely generalized distributed data base management system would reside on a heterogeneous computer network with different data base systems available at various processors. Communication and data transfer would be possible between any nodes in the network. The realization of this goal is still several years in the furture. However, considerable progress in the area of distributed data base systems has been made in both academic and industrial environments.

This report described the principal problem areas in distributed data base management system development. Distributed data base systems share many design problems with both single machine data base systems and computing networks, as well as introducing several new dilemmas.

Recent research in these problem areas is presented to provide a picture of the state of the art of distributed data base development. In addition, the current status of the data base industry with respect to distributed processing is evaluated by reporting the current projects and future plans of selected (anonymous) data base vendors.

A Survey of Developments in Distributed Data Base Management Systems[1]

Technical Report CS77-01

Fred J. Maryanski

Computer Science Department
Kansas State University
Manhattan, Kansas 66506

A

January 1977

[1] This work was supported by the United States Army Computer Systems Command, Grant No. DAAG29-76-G-0108.

This paper will be published in Computer Magazine.

Abstract

Recently we have witnessed the advent of general purpose data base management systems and important advances in computer networks. The combination of the two technologies to produce distributed data base management systems should be the next significant step in commercial systems development. A completely generalized distributed data base management system would reside on a heterogeneous computer network with different data base systems available at various processors. Communication and data transfer would be possible between any nodes in the network. The realization of this goal is still several years in the future. However, considerable progress in the area of distributed data base systems has been made in both academic and industrial environments.

This report describes the principal problem areas in distributed data base management system development. Distributed data base systems share many design problems with both single machine data base systems and computing networks, as well as introducing several new dilemmas.

Recent research in these problem areas is presented to provide a picture of the state of the art of distributed data base development. In addition, the current status of the data base industry with respect to distributed processing is evaluated by reporting the current projects and future plans of selected (anonymous) data base vendors.

INTRODUCTION

Recently, we have witnessed the advent of general purpose data base management systems (DBMS) and important advances in computer networks. The combination of the two technologies to produce distributed data base management systems should be the next significant step in commercial systems development. A completely generalized distributed data base management system would reside on a heterogeneous computer network with different data base systems available at various processors. Communication and data transfer would be possible between any nodes in the network. The realization of this goal is still several years in the future. However, considerable progress in both academic and industrial environments has been made in the area of distributed data base systems.

This report describes the principal problem areas in distributed DBMS development. As indicated by Fry and Sibley[1], distributed data base systems share many design problems with both single machine data base systems and computing networks as well as introducing several new dilemmas.

Recent research in these problem areas is presented to provide a picture of the state of the art of distributed data base development. In addition, the current status of the data base industry with respect to distributed processing is evaluated by reporting the current projects and future plans of selected (anonymous) data base vendors.

Distributed Data Base Taxonomy

Development of a completely general distributed data base management system must evolve through several less complex forms. This evolutionary process is currently in progress. The beginning point of the development is with data base management systems targeted for a single, general purpose computer. The development of such systems is among the most significant events in computer science. For a detailed treatment of the evolution of data base systems, see the work of Fry and Sibley.[1]

In order to properly classify distributed DBMS research and development, several stages in the evolution of a generalized distributed DBMS are listed along with a brief discussion of their current status.

1. DBMS for a single general purpose machine - many such systems are commercially available.

2. Data Base machines - special purpose processors whose function is data management.[2]

3. Back-End DBMS - a network of two or more machines in which one of the processors is dedicated to performing the data base management function. The dedicated data base processor is known as the back-end machine.[3] The back-end machine may be a general purpose computer or contain specialized hardware or firmware.

4. Special purpose distributed DBMS - several special purpose data base systems have been implemented on computer networks. Medical information networks[4,5] have been one of the major areas of emphasis for distributed data systems. Airline reservation systems[6] can also be classified in this area.

5. Single software DBMS on a homogeneous network - this is the next step to be realized in distributed DBMS development. Many hardware manufacturers provide facilities for communication between their own machines. The data base software must be enhanced to allow tasks residing on different processors to communicate. The problems of file allocation, privacy, and deadlock become quite complex at this level.

6. Single software DBMS on a heterogeneous network - by allowing differing brands of processors into the network, compatability problems are introduced. Communication protocols and data conversion schemes are necessary for the extension to heterogeneous networks.

7. Multiple software data base systems on a heterogeneous network- the most general distributed DBMS is one which accomodates users with different data base software systems as well as hardware obtained from multiple vendors. In addition to containing all the problems of the previously mentioned systems, the difficult task of data base translation is introduced. Data base translation entails structural as well as code conversion.

The current status of distributed data base systems reflects the classical relationship of research, development, and production. Research is ongoing in all of the problem areas mentioned in the distributed DBMS breakdown. Progress in the various areas is described in the following sections of this report.

DATA BASE MACHINES AND BACK-END PROCESSORS

The terms "data base machine" and "back-end processor" have recently been added to the computer lexicon. Since they have evolved independently, there is some overlap in their accepted definitions. A data base machine is a special purpose computer dedicated to data base management. A back-end processor is a computer that performs the data management function for one or more different computers. Based upon these definitions, it is possible that a data base machine be used as a back-end processor.

Let us first observe the recent developments in data base machines. One of the most common operations performed in a DBMS is the location of data given some key. A natural method for constructing an efficient

DBMS process is to optimize the data look-up operation. Consequently, associative processors have been proposed for usage as data base machines [7-10].

DeFiore and Bera[8] show that the parallel search capabilities of an associative DBMS allow it to outperform an inverted list structure in an inquiry environment. They have developed a data management system using associative memory at Rome Air Development Center. This system uses the associative processor as a small, fast memory buffer. The data base is stored on conventional secondary storage devices and paged into the associative memory. The associative memory allows searches in the order of microseconds but the page transfer is in the order of milliseconds. The CASSM project at the University of Florida[9] developed the concept of a context-addressable disc system capable of performing data manipulations in secondary memory independent of the CPU. By storing the data base on the associative machine, the excessive loading delay is eliminated. The approach taken in the CASSM project has been followed at the University of Toronto by Ozkarahan, Schuster, and Smith[10] in the design of an associative processor - RAP whose instruction set is a high-level relational data base language. By processing data operations at the machine level, RAP has the potential of overcoming the execution time problems that have hampered relational data base systems. RAP is designed to operate upon data base of up to 100M bits. This capacity can be extended by conventional mass storage devices.

The concept of a processor with data base primitives is also under investigation by Anderson[11] who has proposed placing the data base functions in firmware. The microprogramable data base processor is proposed as a back-end machine by Anderson. This processor could either be directly connected to a single general purpose processor or act as a specialized data base processing node in a network.

The Datacomputer[12] has been developed by Computer Corporation of America as a special purpose data base processor for use in heterogeneous networks. The Datacomputer in its current version is a PDP-10 that provides data base service in the Arpanet. The data base management system which uses the inverted file structure concept is strictly a software implementation. All requests for data base operations are transmitted to the Datacomputer in Datalanguage which is a high-level data base language. Since the Datacomputer interfaces with a variety of machine types, it must perform translations between various physical data representations and logical data structures. Because of its ability to communicate with heterogeneous machines, the Datacomputer as implemented in the Arpanet is the closest existing approximation to a completely distributed data base system. The necessary extensions are the geographic distribution of data which could be accomplished by placing Datacomputers at several points in the network and the incorporation of different software data base systems into the network. The communication between different data base systems remains a major hurdle to a generalized, distributed DBMS.

The INFOPLEX[13] system which is presently under design is intended to be a highly parallel information management system. INFOPLEX is similar in terms of organization to CASSM and RAP in that the data base management function is performed by a complex of microprocessors. Each high-level data base request is functionally decomposed and executed in parallel by the microprocessor complex. The concept of decomposition of data base requests could be applied at a higher level to a network of minicomputer data base machines which share the management of a data base.

The single host, single back-end approach has been implemented as a prototype by Canaday, et al[3] at Bell Telephone Laboratories. The XDMS consisted of a UNIVAC 1108 host which executed the application programs and a META-4 back-end which performed the data base functions. The software system used wasDMS-1100 which is a derivative of the CODASYL specifications.[14,15] XDMS was the first working back-end system and has provided impetus for considerable future work in this area.

Benefits of Back-End DBMS

A back-end DBMS can serve as the nucleus of a distributed data base system. Based upon work presently under way in the data base industry, the author can project the emergence of a back-end computer as a significant product in the near future.

The first versions of the back-end computers will be general purpose minicomputers which contain software data base systems. The back-end machines will interface with IBM 360/370's (or similar mainframes) via conventional linkages. Later versions will include special purpose data base machines and more general and higher speed machine interconnections. These projections are made based upon an analysis of the recent research advances outlined in the previous section and discussions with individuals in the data base industries. An evaluation of the current state of the industry appears in a later section of this paper.

The feasibility of a back-end DBMS in a data processing environment has been the subject of several reports.[3,16-19] These studies indicate that a back-end DBMS can provide benefits in a wide range of areas. The effects of a back-end DBMS are briefly considered here.

a. Performance - This is the critical measure of a system. A back-end computer can overlap its execution with the host machine allowing more computation to take place. Maryanski and Wallentine[20] report the results of simulation studies projecting performance effects of back-end data management systems.

b. Integrity - The overall integrity of a DBMS is enhanced by incorporating a back-end processor into the system since the host and back-end each have the ability to verify the operation of the other machine. Therefore, machine errors can be detected either before or immediately after they impact the data base.

c. Security - If the back-end computer is a data base machine that executes only data base functions and no application programs of any kind, the back-end DBMS can provide a more secure data base.[17] The principal security benefit is that a malevolent programmer cannot construct software which spies on the data base in the back-end. Such a back-end configuration does not provide a solution to all security problems however. The privilege of executing an application program must be carefully guarded and the security of the communication link must be maintained.

d. Economy - A back-end computer can be used to extend the life of a host processor. The acquistition of a back-end minicomputer is an order of magnitude cheaper than an upgrade of a large mainframe. Since a large amount of processing is transferred from the host to the back-end processor, resources on the host computer are made available for additional processing. If the back-end machine is a general purpose minicomputer, it is unlikely that the back-end function will consume all of the machine's resources. Therefore additional computing power becomes available at a relatively low price.

e. Modularity - The basic back-end configuration forms a well-structured compact unit. New machines can be added easily to this configuration either in a multi-processor back-end arrangement or as a stand alone computer.

ORGANIZATION OF DISTRIBUTED DATA BASE SYSTEMS

There have been several alternative organizations proposed for distributed data base systems. The function of the system, the geographic distribution of the data, and the philosophy of the designers all influence the organization of the distributed system. In all cases, the distributed DBMS resides on a computer network. In the next section the characteristics of the interface between the network and data base software are discussed.

Several approaches have been used in the description of distributed data base organization. Aschim[21] classifies data bases according to the geographical distribution of data bases and directories. Data base and directories may either be centralized or distributed under this scheme. Figures 1 and 2 illustrate the situations of distributed data bases with a centralized directory and distributed directory respectively. Aschim also describes the benefits and problems of a single and multiple software systems controlling data base operations.

Booth[22] classifies distributed data bases by the amount of redundancy in the data base. She describes partitioned data bases as logical data base spread across several computers and replicated data bases in which portions of the data base are replicated at different nodes in the network. Data bases may be partitioned based upon accessibility; that is, the locating files at the machine at which they are most likely to receive the heaviest usage. This technique reduces the amount of intermachine communication which can become the limiting factor in distributed data base performance.[20] Booth[22] also describes a vertical partitioning technique, see Figure 3, in which a large central data base is supplemented by several remote data bases.

In an earlier paper,[23] Booth provides an analysis of the tradeoffs between redundancy and division of data in a distributed network. The benefits of redundancy in a distributed environment are increased access to the multiple copies of the data, readily available backup, and decreased communication time to access the data since a copy can be located close to the point at which it is used. The primary problems resulting from redundancy are the cost and complexity of updating a redundant data file and the requirement of additional storage services. When a large active file is divided among several back-end processors in a distributed system, accessibility of the data can be increased. However, there is some control and communications overhead in the distributed situation, as opposed to the case of a single data file accessed by one computer. The development of efficient network software can minimize this overhead.

COMMUNICATION IN A DISTRIBUTED DATA BASE NETWORK

A distributed data base management system must be built upon a computer networking facility. Since communication time is a critical factor in distributed data base performance, an efficient network communciation mechanism is an essential requirement for a distributed data base system. The complexity of the network control software is related to the homogenetity of the machines in the network. If the software data base systems on conversing machines are of different types, then a sophisticated translation mechanism must be developed.

Network control software for distributed data bases has been functionally specified by several researchers. Aschim[21] describes a message switching environment for the communication of data base requests between a host and back-end machine. He describes the information that the host and back-end communication tasks must have available.

According to Aschim,[21] the host communication task must have knowledge of the following items.

1. The identification of the back-end task that will access the requested data;

2. The data base name of the requested data;

3. Translation requirements;

4. The interprocess protocol;

5. The mechanism for interpreting the response.

Similarly, the back-end communication task must have the following information:

1. Type of message received;
2. Translation requirements;
3. The mechanism to satisfy the data base request;
4. Identification of sending process;
5. Conditions for sending a response;
6. Interprocess protocol.

Peebles[24] proposes a separation of the network communication and data base control functions. He specified a network control language for generalized intermachine communication. A translation mechanism is contained within the network control language. A distributed data base management system can be developed on top of the network communication system.

The interface between a data base managment system and a network communication system is explained by Maryanski, et al.[25] The communication system proposed in that paper provides standard methods for inter-task communication and a protocol for the transmission of information among processors. In this system the translation mechanism is considered a part of the data base, not the network, software. Additional details of the application of the network communication system to a distributed data base environment are given by Wallentine and Maryanski.[26]

The Datacomputer[12] relies upon the standard Arpanet communication facilities to exchange information with its host processors. All communication occurs in Datalanguage which can be viewed in this context as a standardized message facility. The Datacomputer performs all translations internally. However, the form of stored data is determined by the host machine.

FILE ALLOCATION IN DISTRIBUTED DBMS

One of the key decisions to be made by a data base administrator is the allocation of data files among physical devices. The ultimate goal of an allocation policy is to equally distribute the utilization of the devices. A file allocation policy requires information, actual or hypothesized, concerning file utilization. In many environments, data base behavior is dynamic. Therefore, utilization patterns must be constantly monitored in order to insure an acceptable file distribution. Naturally, there is a cost associated with monitoring file utilization and reallocating files. A balance between the costs of a non-optimal file organization and the cost of realloca^ion must be achieved.

When a data base is distributed over several machines, the complexity of the file allocation problem increases. In a centralized DBMS, poor file allocation results in heavy traffic on certain channels and excessive waiting for the channels. In a distributed environment, communication cost of obtaining data from a file resident at another network node becomes an important factor. Data must be allocated first among back-end processors and then among the devices attached to the back-end processors. Morgan and Levin[27] have proposed three parameters for describing file allocation algorithms for distributed data base management systems. The parameters are:

1. Level of data sharing;
2. Behavior of access patterns;
3. Type of information available on the behavior of acess patterns.

The level of sharing indicates whether the DDBMS is partitioned, replicated, or some combinationof the two organization schemes. The access pattern of the system may span the spectrum from inquiry only to totalupdate. The important factor for file allocation is if the access pattern may vary significantly. The final parameter is whether the information on the behavior of the access patterns is deterministic or probabilistic.

The standard approach in file allocation problems is to develop a generalized cost equation and then seek a file assignment which minimizes that equation. At the highest level, the cost function can be given as[27]

$$C(A_k) = Q(A_k) + U(A_k) + S(A_k) \qquad (1)$$

where

A_k is the k^{th} assignment;

$Q(A_k)$ is the query cost of the k^{th} assignment;

$U(A_k)$ is the update cost; and

$S(A_k)$ is the storage cost.

As of the present time, the results obtained for file allocation in distributed data base networks are for cases of static, deterministic access patterns. However, Morgan and his associates are studying the properties of networks in which access patterns vary dynamically or are described by probability distributions.[27]

Levin[28] has shown that the multiple file allocation problem for the static, deterministic case can be solved by determining the optimal allocation for the individual files. Levin's cost function is a refinement of eq. (1). The model developed by Chu[29] represents file allocation cost as the sum of storage and transmission times. The overall structure of Levin's and Chu's model are quite close, both involving linear programming techniques to effect a solution.

Casey[30] studies the file allocation problem for tree networks which are a restricted type of network topology. In a tree network, there are no loops formed by internode paths. Casey selected tree networks because of both their simplicity and practicality. Trees have no routing problems and are the optimal organization in a distributed data base with a multiple host, single back-end structure. The most significant feature of tree networks with respect to file allocation is that the cost equations are less complex than those developed by Chu[29] and Levin[28] for more general network organizations.

A different computational approach to computing the optimal file allocation is presented by Casey in another work.[31] He uses a linear cost model for the allocation of network resources that is similar to those used to determine the most economical location for plants and warehouses. A search procedure which is shown to produce an optimal allocation is developed. Casey[31] also suggests some heuristics intended to improve performance of the algorithm. This work, in effect, applies goal-oriented searching techniques commonly used in artificial intelligence work to the file allocation problem.

Chu[32] has also studied the relationship between access type and distributed data base organization. Chu[32] develops cost equations for partitioned, partially replicated, and fully replicated data bases (i.e., a copy of a file at each node which assesses the file). A partially replicated configuration contains one copy of every data file for each cluster in the network. A group of processors joined together via very high speed links (memory-to-memory connections) forms a cluster. Using a cost equation based upon communication, storage, and translation costs, Chu[32] reaches the intuitively appealing result that in query mode a replicated data base provides superior performance,while under heavy update a partitioned data base organization is more efficient. Under the assumption that transmission cost is higher than storage costs, Chu indicates a breakeven point of 10% update between replicated and partitioned and 50% update between fully and partially replicated.

The special case of a 100% query environment is treated by Ghosh[33] who addresses the problem of distributing a data base in order to provide for completely parallel searching. Ghosh provides algorithms that specify a data base distribution that allows parallel searching from a set of queries with a given form and a specific number of processor nodes. He considers both replicated and partitioned data base organization. Again the tradeoff of redundancy in the data base is considered; lower search time versus increased storage costs.

Thus far, the discussion on file organization has concentrated upon the distribution of data among processor nodes of a distributed data base network. Salasin[34] describes a method for distributing a data base over the storage hierarchy of a single processor. The storage devices of the processor are ordered in terms of access speed. The difference between the conventional storage organization and Salasin's hierarchical approach is illustrated in Figure 4. The most significant performance feature of this proposed storage arrangement is the bufferring of data. If data is found at level K, it is also present at all higher levels. Salasin constructs probabilistic models which indicate that bufferring provides performance benefits for sequential, random, and linked list file organizations.

File organization has been studied more thoroughly than other subjects related to distributed data bases. Many linear programming methods have been proposed for the determination of optimal file placement. The practicality of these methods for use in data processing environments is a matter of conjecture. The main problems are that the algorithms require precise usage statistics and do not consider constraints imposed by security or company policy.

DEADLOCKS

A good file allocation scheme results in efficient utilization of a distributed data base management system. However, if a distributed

DBMS designer ignores file allocation, it is likely that the system will still operate (perhaps in a very inefficient manner). This situation does not hold for the deadlock problem. If the system designer does not consider the possibility of deadlocks, severe problems may arise.

Deadlock in a DBMS is an unfortunate side effect of the need for a portion of the data base to be shared by several data base tasks, at least one of which is updating the shared information. For example, if a record is to be updated by a task, it is necessary that no other task be allowed to access the record during the update procedure. Failure to provide a blocking mechanism can result in incorrect information appearing in the data base. If there are portions of the data base that may be accessed simultaneously by several tasks, then a deadlock condition may occur. Deadlock occurs when two or more tasks have blocked each other from execution by locking shared portions of the data base. Figure 5 illustrates deadlock of two data base tasks.

The underlying cause of deadlock in data base systems is the organization of conventional secondary storage media. In order to optimize the utilization of secondary storage, several requests must be processed simultaneously. This organization tends to reduce disk head movement and latency which is often the limiting factor in performance of a data base system. The deadlock problem can be avoided in systems which do not use convential secondary storage. Associative machines [7-10] batch all data base requests and provide the task issuing the request exclusive control of the data base, thus avoiding deadlock.

The deadlock problem has been studied at great length by researchers in operating systems. The general principles for the detection or prevention of deadlock that have been developed by these researchers are also applicable to data base management. However, the DBMS deadlock problem is compounded by the need to insure that uniformly correct data is maintained throughout the system.

For example, if in the situation depicted in Figure 5, the deadlock is resolved by returning Task A to its starting point and releasing its resources (this procedure is called "rollback"), then Record 1 must be restored to its condition prior to being modified by Task A. If the locking procedure used permits retrieval while preventing updates by other tasks, then it is possible that some Task C may have retrieved and operated upon Record 1 in its altered state (after step a_3). In this situation Task C would also have to be rolled back. Rollback of Task C may result in the necessity of rolling back still other tasks in the system, thus causing a substantial performance degradation.

Deadlock in data base management systems has been investigated by several researches.[35-41] However, relatively little work has concentrated on deadlock in distributed data base management systems. In a distributed DBMS the problem is complicated by the fact that the interacting tasks may reside upon different machines. Therefore, the communication overhead in the rolling back of tasks can become very substantial. Consequently, in a distributed DBMS, a deadlock prevention scheme may provide better overall system performance than a deadlock detection approach.

Preventing deadlocks in a distributed DBMS is the subject of a report by Chu and Ohlmacher.[42] They propose two approaches to deadlock prevention. The first method requires a data base task to indicate its resource (file) requirements before initiation. A task is started only if all requested resources can be assigned to the task. This approach is very straightforward. However, a task may not access all files in a given execution. Thus, the initiation of a task may be needlessly delayed.

The second technique for deadlock prevention proposed by Chu and Ohlmacher is based upon the notion of task sets. A task set is a collection of tasks with access to common files. Whenever a task has the need to access a file, the system determines if all files that may be accessed by the requesting task and other members of its task set are available. If all such files are free, the task is permitted to proceed. Otherwise, the task must wait until its files are available.

The task sets change as tasks are initiated and terminated. Chu and Ohlmacher present an algorithm for assigning the control of a particular task set to a processor node in the network. All requests for files by tasks in the set must be directed to the controlling processor.

An analysis of the two proposed deadlock prevention algorithms for distributed data base systems, indicates that the dynamic nature of the second technique has both benefits and drawbacks. It has the advantage of allowing tasks to proceed until an actual file request occurs. However, the maintenance of the task sets may require considerable communication overhead in a network environment.

An important factor for any proposed deadlock handling algorithm for distributed data bases would be operational efficiency. Due to the complexity of distributed data base systems, it is difficult to determine the efficiency of such algorithms analytically. Therefore, the practicality of the treatment of a deadlock in a distributed DBMS cannot be determined until more distributed data bases are implemented.

DATA TRANSLATION

One of the problems that faces the designer of a distributed DBMS composed of multiple software systems on a heterogeneous network is data incompatibility. The problem of disparate internal data representations is complicated by different logical structures in the data base system. Since these differences are a fact of life in data processing, a method of data base translation is necessary for the most general case of distributed network. For a network with K different data base systems, a "brute force" translation approach is to construct a unique translator for each pair of data base systems. The problem of translating between two given data base systems on specified computers is a well defined but non-trivial task. However, this approach would require each DBMS node to have 2 (K-1) translators available to map to and from every other DBMS node in the network.

Several alternatives to the "brute force" or "K to K" data translation approach have been considered. A major effort in this area is the University of Michigan Data Translation Project.[43-45] Figure 6 illustrates the translation methodology developed by Fry and his associates at Michigan. All data bases are described using a universal Stored Data Definition Language (SDDL). The translations are driven by tables produced by compilers for the SDDL and the Translation Definition Language (TDL). The TDL is employed to express the relationship between the source and target data bases. This translation methodology requires only one translation program at each DBMS node. However the translator must be supplemented with an SDDL Table and (K-1) TDL Tables. Birss and Fry[45] discuss the feasibility of the Data Translation Project Methodology based upon their prototyping experiences.

Schneider[46] proposes a Data Specification and Conversion Language (DSCL) for data base networks. The DSCL is a high level language for data translation. Schneider suggests that one machine in the distributed DBMS serve as the network translator as shown in Figure 7. All communication with the translation machine uses DSCL. The translation machine effectively contains K^2 different translation programs. The main advantage of this technique is that the only additional software required on the DBMS nodes are utility routines to map to and from DSCL.

The Datacomputer[12] which is a back-end machine in the ARPA network communicates with its host machines only in Datalanguage. Each host processor accessing information controlled by the Datacomputer must perform translations to and trom the Datalanguage format. The Datacomputer is a special case of the general data translation problem. However, the fact that it is a viable node in the ARPA network illustrates the feasibility of that approach.

The approaches to data translation discussed thus far are totally automatic. Su and Lam[47] have designed an interactive translation system in which the user participates in composing the logical structure of the target data base. The system requires a separate translator for each distinct target machine. This approach provides the user with considerable flexibility in restructuring the target data base. However, it requires user intervention for any intermachine data base communication. Therefore, the interactive approach is best suited for infrequent large scale data transfers. A prototype has been constructed by Su and Lam.

The art of generalized data translation is currently at the prototype stage. Progress (or lack of it) in data translation is one of the limiting factors in the development of heterogeneous distributed data base systems. As indicated by Fry and Deppe,[48] the two major problems facing workers in the data translation area are structural translation, several approaches to which we have discussed here,[43-47] and query translation which has been addressed to a limited extent in the Datacomputer project.[12] The development and wide acceptance of standardized data base systems would of course reduce the need for structural translation. This is the intent of the CODASYL proposals[14,15] and would also be a benefit of relational data base systems.[49]

SECURITY

Distributed data base systems with information and control spread over a number of computers pose many interesting security problems. Since many data bases contain classified or private information, thought must be given to the security of data before determining if any benefits are to be gained by multi-computer access. The principal security question with regard to distributed data bases is whether a distributed DBMS is inherently more or less secure than a single machine system. An analysis of distributed data base systems indicates that they provide both advantages and drawbacks with respect to security.

The security benefits arise in configurations which contain dedicated back-end machines. Lownethal[17] indicates that a computer solely dedicated to the processing of data base operations is able to screen every data base request from the host machines. One important security aspect of the dedicated back-end machine is that no application programs execute on it. This eliminates the threat of a malevolent program monitoring data base activity.

The most outstanding security liability of a distributed data base system is the use of public communication lines in geographically dispersed networks. An intruder using current technology can easily monitor the transmissions between remote installations. A designer of a secure geographically distributed DBMS must rely upon encryption techniques to preserve security.

STATE OF THE INDUSTRY

Distributed data bases are on the verge of becoming a commercial reality. Back-end data base systems are presently in the development stage at the installations of several hardware and software vendors. It is difficult to report industrial progress without either providing free publicity or revealing proprietary information. However, it is important that the significance of distributed data base systems be emphasized by describing the work underway in the commercial sector. Therefore, this section contains brief descriptions of projects currently underway at several vendors' installations with no specific references to either the vendor or its product line.

The back-end machine has been a focal point of the industry's thrust into distributed systems. Virtually all vendors have followed Canaday's[3] initial prototype by using a minicomputer as the back-end machine. One software vendor is presently developing a back-end DBMS targeted for a specific large mainframe host and minicomputer back-end Their DBMS software presently operates on both machines in a stand alone manner. The major developmental effort on the project is the implementation of a communication system between the two machines.

When complete, this particular back-end DBMS will be suitable for use with both locally and remotely connected machines.

Another software vendor is taking the data base machine approach to distributed data base systems. This company is developing a back-end DBMS version of their software on a minicomputer and a generalized communication system written in an easily portable systems implementation language. They intend to market the minicomputer as a "blackbox" which can be attached by means of their communications software to several large mainframe computers..

Hardware vendors are also actively pursuing the idea of distributed data bases. One manufacturer is studying the feasibility of constructing a back-end machine from a cluster of microcomputers in a manner similar to Madnick's INFOPLEX.[13] In the system being studied, a data base request would be decomposed into primitives and processed in parallel by the microcomputers. This approach is very well suited for a relational DBMS.[50]

Another hardware manufacturer is investigating the concept of a microprogrammable back-end machine. This back-end processor would be a minicomputer that could be connected via a high speed interface to the vendor's large mainframe CPU's. Through the use of special purpose data base instructions in firmware, a high performance back-end machine can be developed. The vendor is also considering configurations of multiple back-end processors.

From this small sample of the activity in the data base industry, it can be seen that distributed data base systems, in particular back-end machines, will soon appear in the marketplace. This is a result of an intersection of advances in hardware and software technology with a need for greater access to and sharing of information.

CONCLUSION

Distributed data base management systems are the focus of a large amount of research and development activity in both the academic and industrial environments. Distributed data base systems provide a means of extending the capacities of computing systems to allow a wide range of information to be accessed by many people. As is typical of the current computer environment, the hardware technology of distributed data base has advanced further than the software.

Many obstacles remain before a truly general distributed DBMS will appear. However, back-end data base systems, data base machines, and distributed information systems are available now (or will be available within the next year). In the next several years, advances in distributed data base management systems should be among the most significant in the computer field.

REFERENCES

1. J. P. Fry and E. H. Sibley, "Evolution of Data-Base Management Systems", *Computing Surveys*, Vol. 8, No. 1, Mar. 1976, pp. 7-42.

2. R. I. Baum and D. K. Hsiao, "Database Computers—A Step Towards Data Utilities", IEEE *Trans. on Computers*, Vol. C-25,No. 12, Dec. 1976, pp. 1254-1259.

3. R. E. Canaday, et al., "A Back-End Computer for Data Base Management", CACM, Vol. 17, No. 10, Oct. 1974, pp. 575-582.

4. E. Chang, "A Distributed Medical Data Base", *Computer Networks* Vol. 1, No. 1, June 1976, pp. 33-38.

5. G. Wiederhold, J. F. Fries, and S. Weyl, "Structured Organization of Clinical Data Bases", *Proc. AFIPS National Computer Conference*, Vol. 44, May 1975, pp. 479-485.

6. J. Knight, "A Case Study-Airline Reservation Systems", *Proc. IEEE*, Vol. 60, No. 11, Nov. 1972, pp. 1423-1431.

7. P. B. Berra, "Some Problems in Associative Processor Applications to Data Base Management", *Proc. AFIPS National Computer Conference*, Vol. 43, June 1974, pp. 1-5.

8. C. R. DeFiore and P. B. Berra, "A Data Management System Utilizing an Associative Memory", *Proc. AFIPS National Computer Confernece*, Vol. 42, June 1973, pp. 181-185.

9. S. Y. Su, G. P. Copeland, Jr., and G. J. Lipovski, "Retrieval Operations and Data Representations in a Context-Addressable Disk System", *Proc.* *ACM SIGPLAN-SIGIR Interface Meeting*, Nov. 1973, pp. 144-160.

10. E. A. Ozkarahan, S. A. Schuster, and K. C. Smith, "RAP--An Associative Processor for Data Base Management", *Proc. AFIPS National Computer Conference*, Vol. 44, May 1975, pp. 379-387.

11. D. R. Anderson, "Data Base Processor Technology", *Proc. AFIPS National Computer Conference*, Vol. 45, June 1976, pp. 811-818.

12. T. Marill and D. Stern, "The Datacomputer--A Network Data Utility", *Proc. AFIPS National Computer Conference*, Vol. 44, May 1975, pp. 389-395.

13. S. E. Madnick, "INFOPLEX--Hierarchical Decomposition of a Large Information Management System using a Microprocessor Complex", *Proc. AFIPS National Computer Conference*, Vol. 44, June 1975, pp. 581-586.

14. *CODASYL COBOL Journal of Development*, Dept. of Supply and Services, Material Data Management Branch, Ottawa, Ontarior K1A OS5. (revised to) June 1976

15. COSASYL Data Description Language Journal of Development, Document C1362:113, U.S. Government Printing Office, Washington, D.C., 1973.

16. G. C. Everest, "The Futures of Database Management", Proc. ACM SIGMOD Workshop, May 1974, pp. 445-462.

17. E. I. Lowenthal, "The Backend Computer", MRI Systems Corp., P.O. Box 9968, Austin, Texas 78766, Apr. 1976.

18. F. J. Maryanski, P. S. Fisher, and V. E. Wallentine, "Evaluation of Conversion to a Back-End Data Base Management System", Proc. ACM Annual Conference, Oct. 1976, pp. 293-297.

19. K. M. Whitney, "Fourth Generation Data Management Systems" Proc. AFIPS National Computer Conference, Vol. 42, June 1973, pp. 239-244.

20. F. J. Maryanski and V.E. Wallentine, "A Simulation Model of a Back-End Data Base Management System", Proc. Pittsburgh Modeling and Simulation Conference, Apr. 1976, pp. 252-257.

21. F. Aschim, "Data Base Networks-An Overview", Management Informatics, Vol. 3.1, Feb. 1974, pp. 12-28.

22. G. M. Booth, "Distributed Information Systems", Proc. AFIPS National Computer Conference, Vol. 45, June 1976, pp. 789-794.

23. G. M. Booth, "The Use of Distributed Data Bases in Information Networks", Proc. 1st International Conference on Computer Communication: Impacts and Implications, Oct. 1972, pp. 371-376.

24. R. Peebles, "Design Considerations for a Distributed Data Access System", Ph.D. Dissertation, Moore School of Electrical Engineering, University of Pennsylvania, May 1973.

25. F. J. Maryanski, et al., "A Minicomputer Based Distributed Data Base Management System", Proc. NBS-IEEE Trends and Applications Symposium: Micro and Mini Systems, May 1976, pp. 113-117.

26. V. E. Wallentine and F. J. Maryanski, "Implementation of a Distributed Data Base System", TR CS 76-03, Dept. of Computer Science, Kansas State University, Manhattan, Ks 66506, Feb. 1976.

27. K. D. Levin and H. L. Morgan, "Optimizing Distributed Data Bases--A Framework for Research", Proc. AFIPS National Computer Conference, Vol. 44, June 1975, pp. 473-478.

28. K. D. Levin, "Organizing Distributed Data Bases in Computer Networks", Ph.D. Thesis, The Wharton School, University of Pennsylvania, Sept. 1974.

29. W. W. Chu, "Optimal File Allocation in a Multiple Computer System", IEEE Trans. on Computers, Vol. C-18, No. 10, Oct. 1969, pp. 885-889.

30. R. G. Casey, "Design of Tree Networks for Distributed Data", Proc. AFIPS National Computer Conference, Vol. 42, June 1973, pp. 251-257.

31. R. G. Casey, "Allocation of Copies of a File in an Information Network", Proc. AFIPS Spring Joint Computer Conference, Vol. 40, 1972, pp. 617-625.

32. W. W. Chu, "Performance of File Directory Systems for Data Bases in Star and Distributed Networks", Proc. AFIPS National Computer Conference, Vol. 45, June 1976, pp. 577-587.

33. S. P. Ghosh, "Distributed Data Base with Logical Associations on a Computer Network for Parallel Searching", RJ 1439 (#22109), IBM Research Lab, San Jose, Ca. 95193, Aug. 1974.

34. J. Salasin, "Hierarchical Storage in Information Retrieval", CACM, Vol. 16, No. 5, May 1973, pp. 291-295.

35. A. Shoshani and A. J Bernstein, "Synchonization in a Parallel Accessed Data Base", CACM, Vol. 12, No. 11, Nov. 1969, pp. 604-607.

36. A. J. Collmeyer, "Database Management in a Multi-Access Environment", Computer, Vol. 4, No. 6, Nov. 1971, pp. 36-46.

37. J. E. Shemer and A. J. Collmeyer, "Database Sharing: A Study of Interference, Roadblock and Deadlock", Proc. ACM SIGFIDET Workshop, Nov. 1972, pp. 147-163.

38. E. Yourdon, The Design of On-Line Computer Systems, Prentice-Hall, Englewood Cliffs, N. J., 1972, pp. 310-353.

39. P. F. King and A. J. Collmeyer, "Database Sharing--An Efficient Mechanism for Supporting Concurrent Processes", Proc. AFIPS National Computer Conference, Vol. 42, June 1973, pp. 271-275.

40. G. C. Everest, "Concurrent Update Control and Database Integrity", in Data Base Management. J. W. Klimbie and K. L. Koffeman (eds.), North-Holland, Amsterdam, Apr. 1974, pp. 241-270.

41. K. P. Eswaran, et al., "The Notions of Consistency and Predicate Locks in a Data Base System," CACM, Vol. 19, No. 11, Nov. 1976, pp. 624-633.

42. W. W. Chu and G. Ohlmacher, "Avoiding Deadlocks in Distributed Data Bases," Proc. ACM Annual Conference, Nov. 1974, pp. 156-160.

43. K. Yamaguchi and A. G. Merten, "Methodology for Transferring Programs and Data", Proc. ACM SIGMOD Workshop, May 1974, pp. 141-155.

44. A. G. Merten and J. P. Fry, "A Data Description Language Approach to File Translation", Proc. ACM SIGMOD Workshop, May 1974, pp. 191-205.

45. E. W. Birss and J. P. Fry, "Generalized Software for Translating Data", *Proc. AFIPS National Computer Conference*, Vol. 45, June 1976, pp. 889-897.

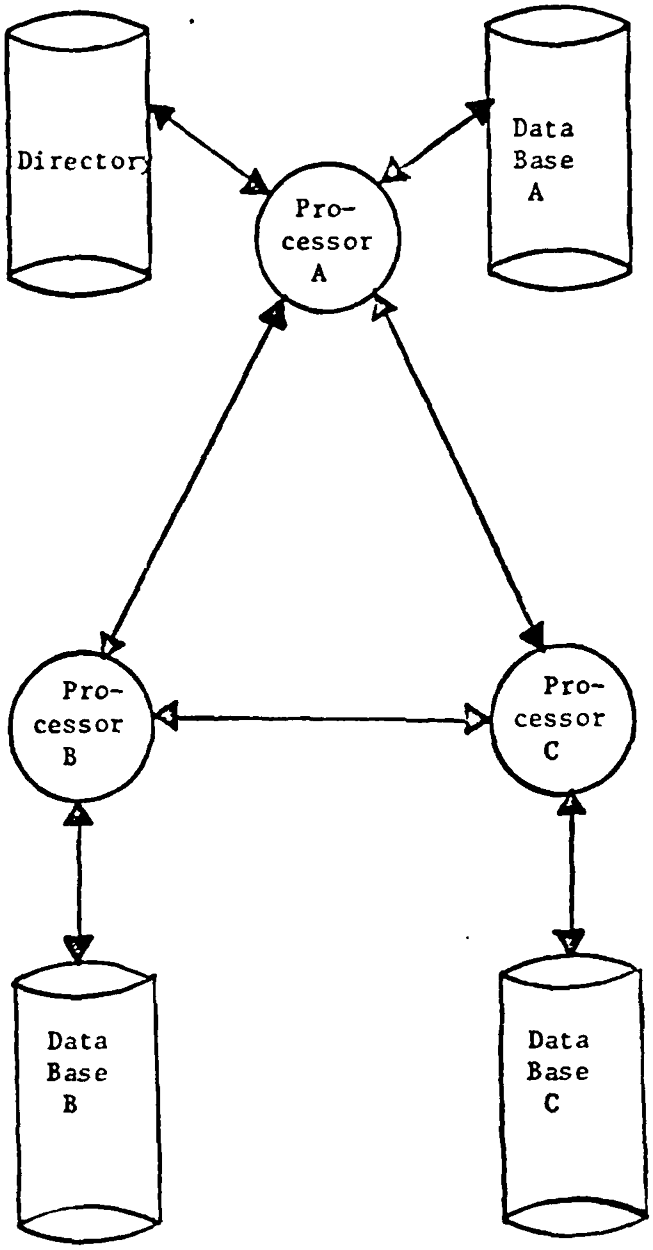
46. G. M. Schneider, "DSCL--A Data Specification and Conversion Language for Network", *Proc. ACM SIGMOD Workshop*, May 1975, pp. 139-148.

47. S.Y.W. Su and H. Lam, "A Semi-Automatic Data Translation Scheme for Achieving Data Sharing in a Network Environment", *Proc. ACM SIGMOD Workshop*, May 1974, pp. 227-247.

48. J. P. Fry and M. E. Deppe, "Distributed Data Bases: A Summary of Research", *Computer Networks*, Vol. 1, No. 2, 1976.

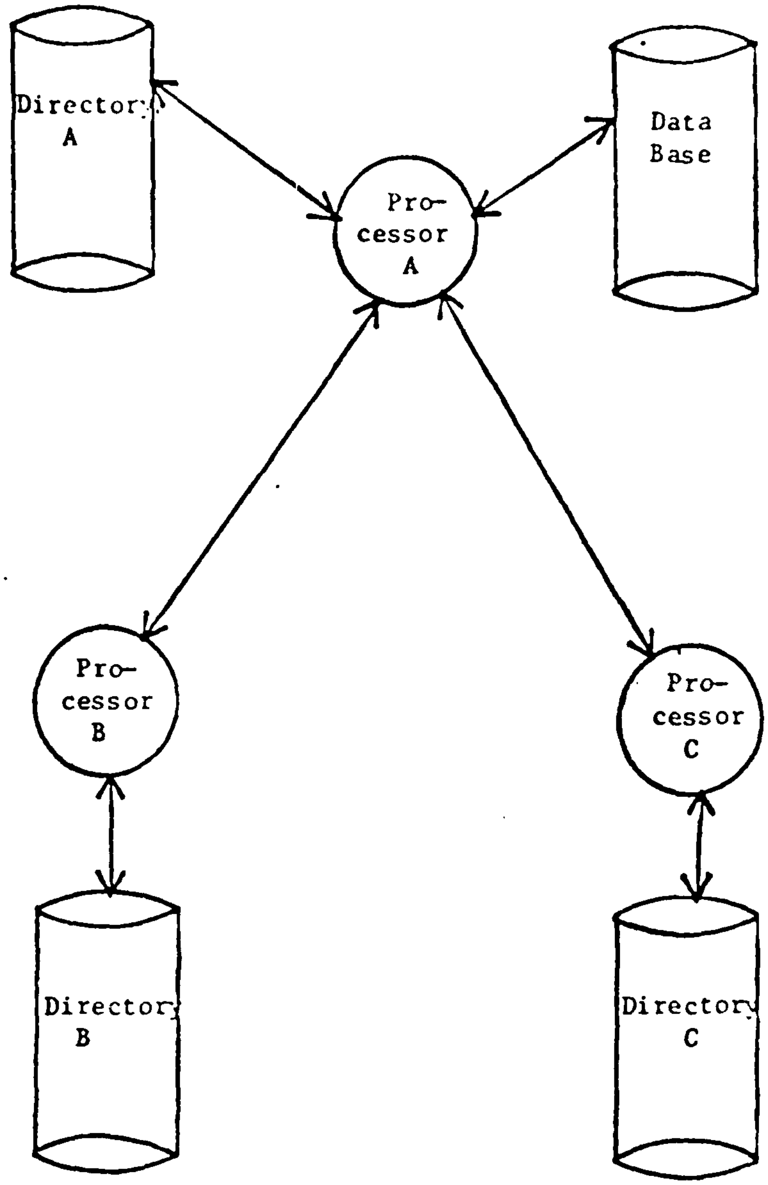
49. D. D. Chamberlain, "Relational Data Base Management Systems", *Computing Surveys*, Vol. 8, No. 1, Mar. 1976, pp. 43-66.

50. P. Y. Chang, "Parallel Processing and Data Driven Implementation of a Relational Data Base System," *Proc. ACM Annual Conference*, Oct. 1976, pp. 314-318.



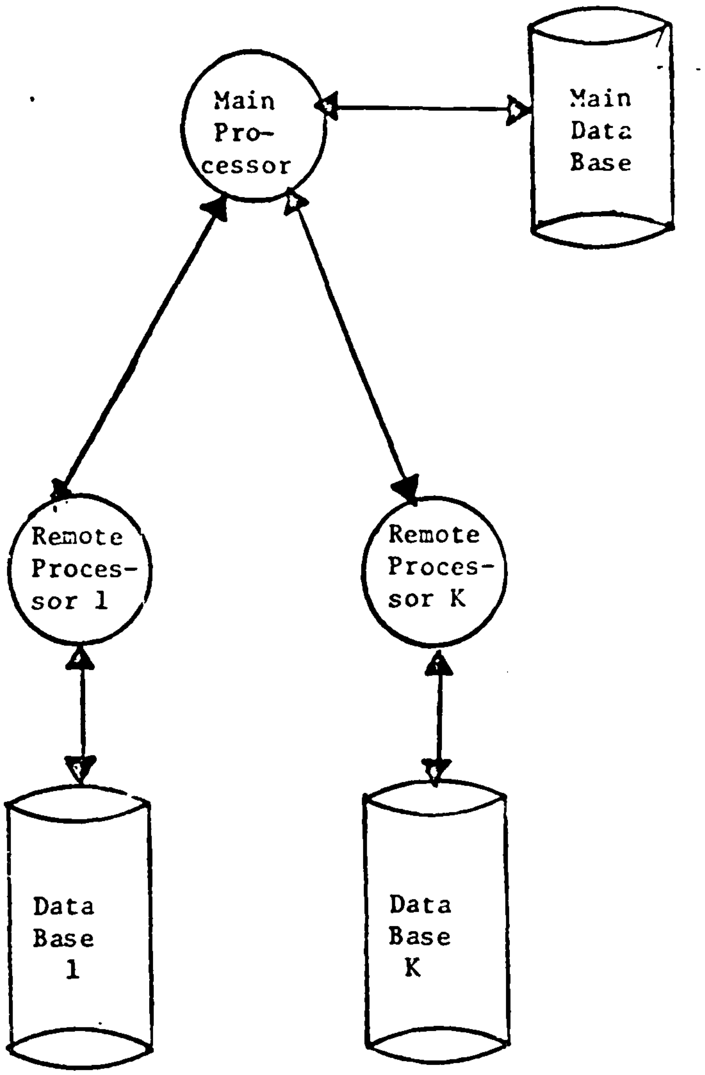
Distributed Data Bases, Central Directory

Figure 1



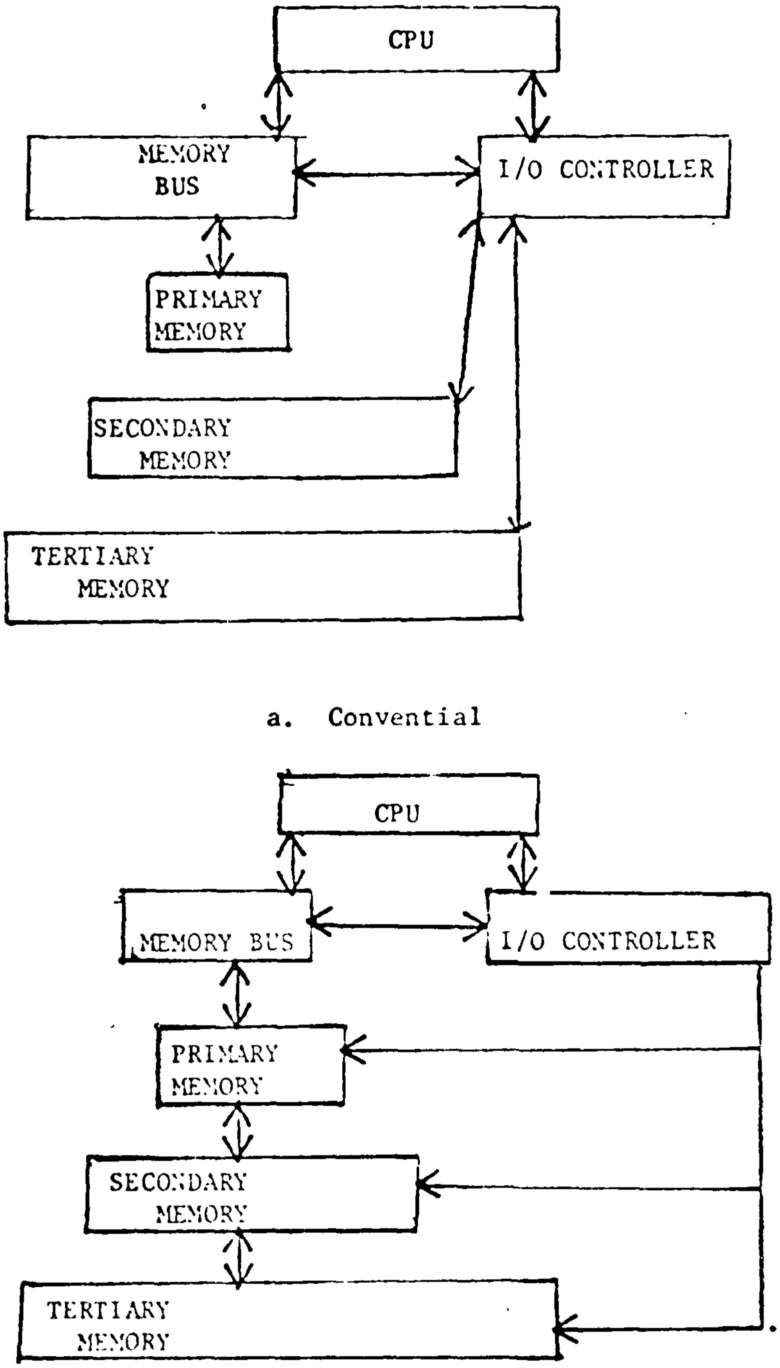
Centralized Data Base, Distributed Directories

Figure 2



Vertically Partitioned Data Base

Figure 3



a. Convential

b. Hierarchical

Conventional and Hierarchical Storage Organization

Figure 4

Task A

Step	Action
a_o	Request Record 1
a_1	Receive Record 1
a_2	Lock Record 1
a_3	Modify Record 1
a_4	Request Record 2
a_5	Receive Record 2
a_6	Lock Record 2
a_7	Modify Record 2
a_8	Release Record 1
a_9	Release Record 2

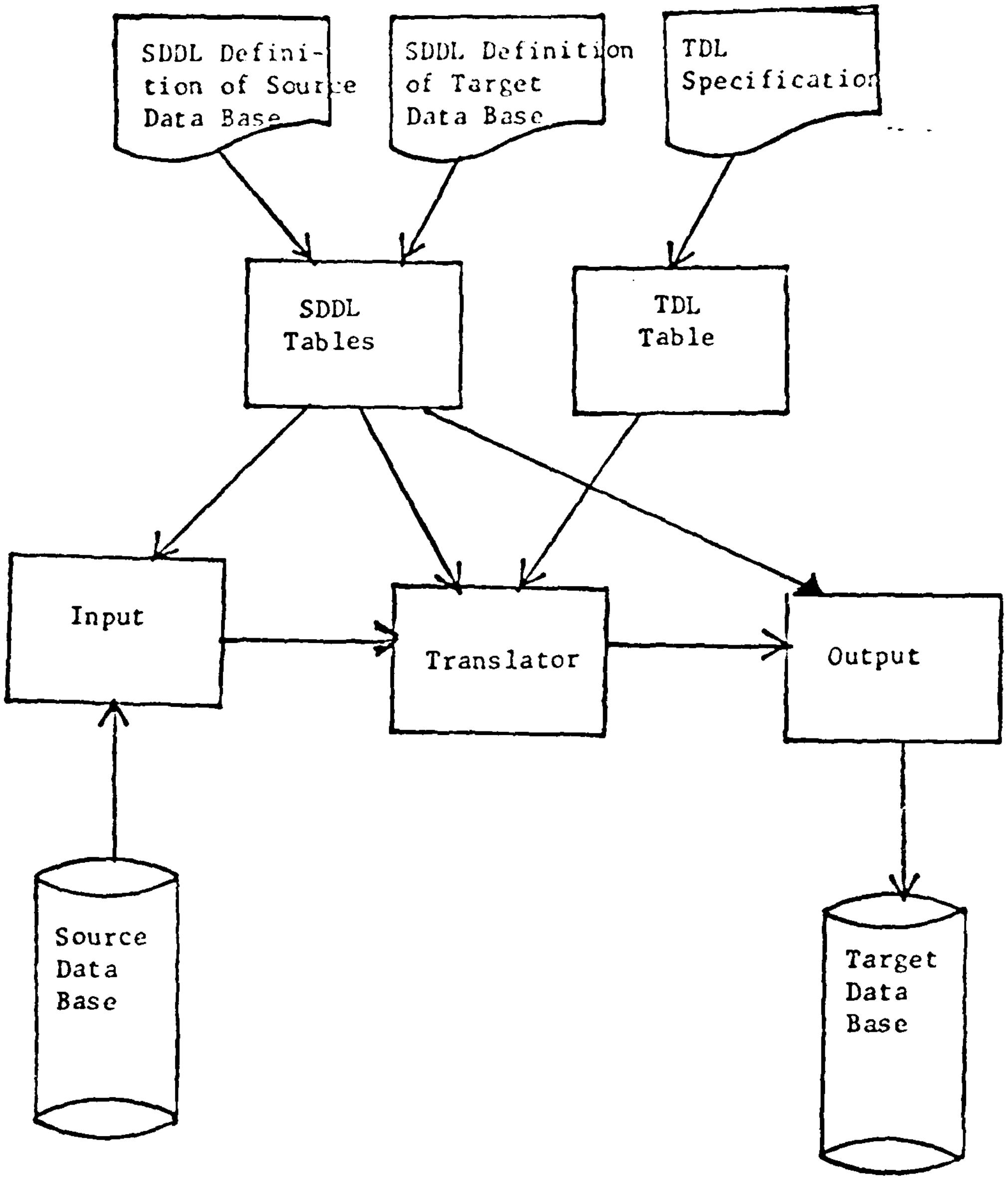
Task B

Step	Action
b_o	Request Record 2
b_1	Receive Record 2
b_2	Lock Record 2
b_3	Modify Record 2
b_4	Request Record 1
b_5	Receive Record 1
b_6	Lock Record 1
b_7	Modify Record 1
b_8	Release Record 2
b_9	Release Record 1

Sequence of Steps Leading to Deadlock

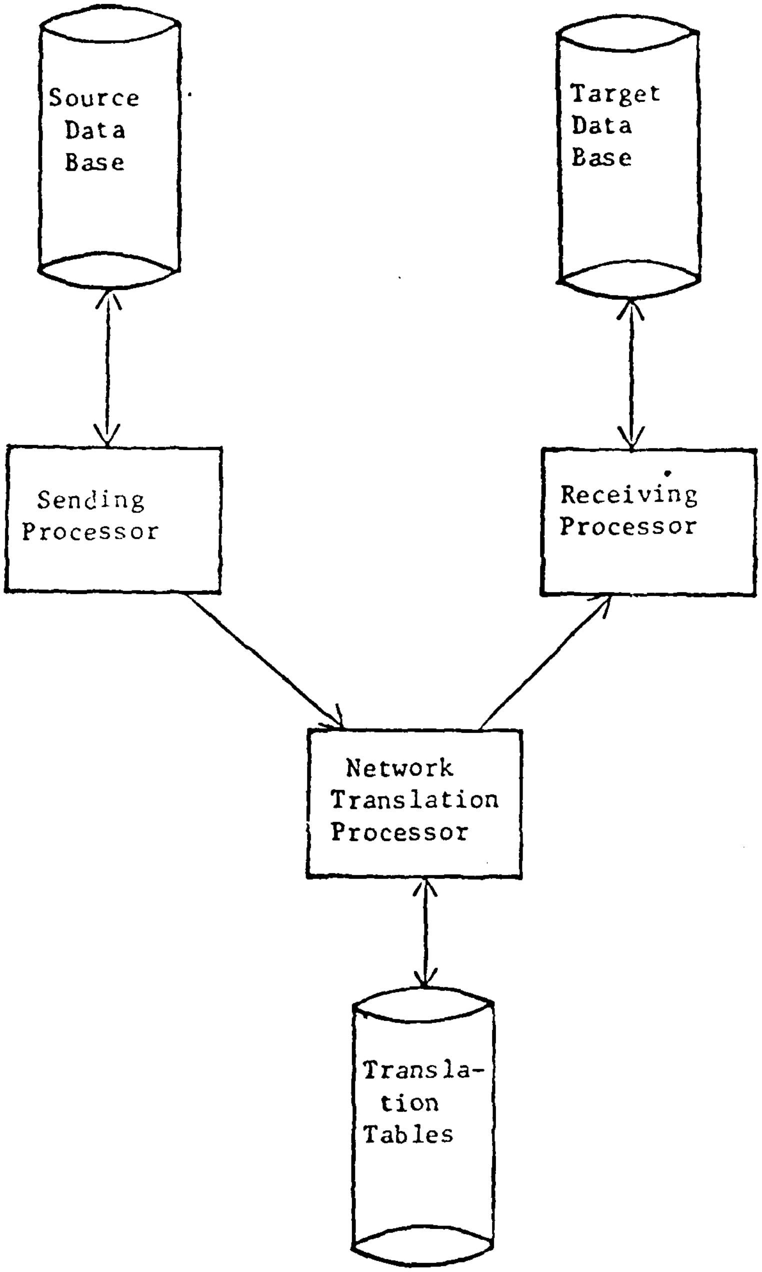
$a_o \quad b_o \quad a_1 \quad a_2 \quad b_1 \quad b_2 \quad a_3 \quad b_3 \quad a_4 \quad b_4$

DBMS Deadlock
Figure 5



U. of Michigan Translation Approach

Figure 6



DSCL Network Organization

Figure 7